

TAREAS APACHE

Cousas a ter en conta

Na máquina virtual correspondente xa están instalados algúns paquetes necesarios, entre os que se atopan principalmente docker e docker-compose.

A IP da máquina en cuestión é 192.168.56.80, podedes modificala para adaptala á que vos queirades.

Ademais atópanse os arquivos Dockerfile e docker-compose.yml nos que se especifican as directivas e servizos a crear, estes pódense modificar para adaptalos ao escenario da tarefa en concreto.

Recordatorio

Dockerfile

GNU nano 3.2	Dockerfile	Modificado
<pre>#Indicamos a imaxe da que se parte FROM debian #Instalamos paquetes de utilidades RUN apt update && apt install -y vim iputils-ping nano procps apache2 php git #Creamos o directorio /var/www/html/web RUN mkdir /var/www/html/web #Indicamos o porto no que vai escoitar o noso contedor EXPOSE 80 #Comando para arrancar ssh y entrar en bucle de espera ENTRYPOINT ["apache2ctl", "-D", "FOREGROUND"]</pre>		

docker-compose.yml

```
GNU nano 3.2 docker-compose.yml

version: '3'
services:
  #Service apache toma el Dockerfile del directorio ./apache
  apache:
    build: .
  #Mapeo de puerto 8080 del host al 80 del container
  ports:
    - "8080:80"

  #Mapea el volumen apache_data al directorio de datos de apache
  volumes:
    - ./web:/var/www/html/web
    - ./srv:/home/srv
```

Lembrar que a versión é únicamente un elemento informativo.

Na directiva **services** estamos a crear os nosos servizos. Neste caso creamos un, ao que lle damos o nome de apache, este nome non é máis que unha etiqueta, poderíadeslle poñer outro calquera.

Dentro do servizo, na directiva **build** temos que indicar a ubicación do Dockerfile (se o Dockerfile se atopa no mesmo directorio que o docker-compose.yml indícase cun punto, noutro caso poñeríamos o directorio en cuestión).

Na directiva ports asociamos un porto da máquina anfitrión asociado ao porto no que escoita o contedor, respectivamente, como aparecen na imaxe, ambos separados por :

Por último, como dixemos os contedores son volátiles, polo que, para manter os datos unha vez pechado o contedor, configúranse os volumes. Para iso asociamos un directorio na máquina anfitrión no que se van gardar os datos xerados no directorio que indiquemos no contedor. Neste caso **./web** fai referencia ao directorio na máquina anfitrión (que se xera de maneira automática) e **/var/www/html/web** e aquel correspondente na máquina anfitrión por debaixo do cal se gardarán os datos xerados. Pódense crear os volumes que se consideren en función dos datos que se queiran conservar.

Lanzamento do contedor

Unha vez xerados ambos arquivos debemos lanzar o noso contedor.

O primeiro será colocarnos do directorio onde se atopa o `docker-compose.yml` e empregar os comandos:

docker-compose up -d (crea e inicia os contedores para todos os servizos especificados)

docker-compose exec apache bash (executa o servizo apache no contedor abrindo a consola de comandos para poder interactuar con ela)

Outros comando de docker a ter en conta

docker stop nome_contedor (parar contedor)

docker rm nome_contedor (eliminar contedor)

docker ps (ver listado de contedores activos, aquí tamén se da información do nome, a id, etc. do contedor)

docker ps -a (para ver listado de todos os contedores (activos e inactivos))

```
root@base:~# ls
docker-compose.yml Dockerfile srv web
root@base:~# docker-compose up -d
Creating root_apache_1 ... done
root@base:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
85bd1bff5008   root_apache    "apache2ctl -D FOREG..." 17 seconds ago Up 15 seconds 0.0.0.0:8080->80/tcp, :::8080->80/tcp root_apache_1

root@base:~# docker-compose exec apache bash
root@85bd1bff5008:/#
```

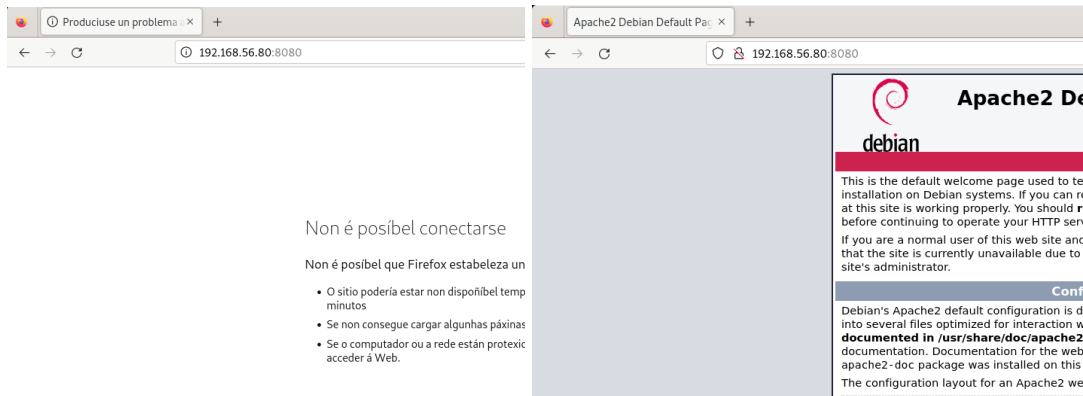
Vemos como se xera o directorio web como se indicou no Dockerfile

```
root@85bd1bff5008:/# ls /var/www/html
index.html web
```

Ademais, ao ter volumes configurados os arquivos que se foron gardando poden ser recuperados, polo que hai arquivos por debaixo deste directorio.

```
root@85bd1bff5008:/# ls /var/www/html/web
fotos server1 server2
```

Vamos agora a comprobar que está a funcionar. Como apache xa trae unha configuración por defecto poderíamos conectar directamente coa IP no navegador, lembrando sempre incluír o porto (8080, xa que é o da máquina anfitrión) separado por dous puntos. Lembrar sempre estar executando o contedor, xa que noutro caso daría erro (imaxe esquerda).



O que deberíamos ver é a imaxe da dereita, o que indica que todo está a funcionar.

Pero ¿qué é o que estamos vendo? Por defecto, ao instalar apache temos un VirtualHost xa configurado.

No directorio **sites-available (/etc/apache2/sites-available)**

```
root@85bd1bfff5008:/# ls /etc/apache2/sites-available
000-default.conf  default-ssl.conf
```

Polo que ao que estamos accedendo é ao sitio virtual configurado nese arquivo

```

GNU nano 7.2 /etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>

```

O **ServerAdmin** indica o enderezo de contacto que o servidor inclúe nas mensaxes de erro que se devolven ao cliente.

O **DocumentRoot** é o directorio desde o cal o demo httpd serve os ficheiros. No caso de ter un arquivo index, sen necesidade de indicalo o servidor da ese arquivo de maneira directa. Deste xeito o arquivo que se nos amosa en pantalla é o index.html presente no directorio **/var/www/html**

```

root@85bd1bfff5008:/# ls /var/www/html
index.html  web

```

Nun primeiro apartado da terefa pídesenos “Configura dous sitios virtuais (server1 e server2). Cada un co seu DocumentRoot en dous cartafos diferentes (montados coma volumes colgando de /web”

Neste caso queremos traballar con VirtualHost baseados en nome, é dicir, temos múltiples sitios web no mesmo enderezo.

A configuración a incluír sería a seguinte:

```
<VirtualHost *:80>
    ServerName server1.rlinares.dedyn.io
    ServerAlias server1
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1
</VirtualHost>
```

Onde o * indica que o mapeado baseado no enderezo IP irrelevante, **ServerName** é o nome do host (hostname) que o servidor (ou VirtualHost) usa para identificarse a si mesmo e **ServerAlias** son nomes alternativos para un host para empregar host virtuais baseados en nomes.

Para realizar isto imos facer unha copia do arquivo por defecto **000-default.conf**, e incluír nesta copia o noso sitio virtual. A idea de modificar a copia e non a orixinal é a de ver o feito de que todos os arquivos de configuración que se creas en **sites-available** teñen que ser activados para que comeces a funcionar, xerando unha ligazón simbólica a este arquivo de configuración (**a2ensite nome_arquivo**) no directorio **sites-enabled**.

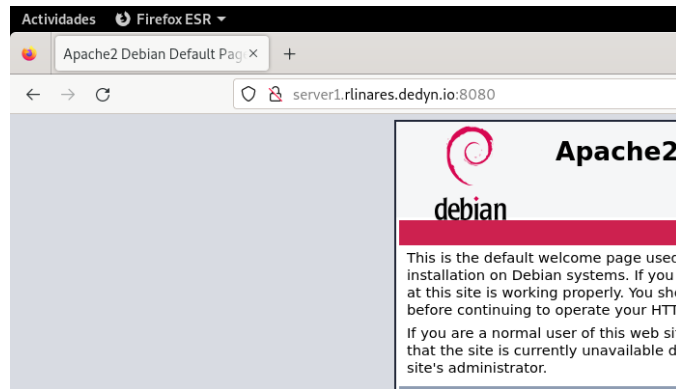
Facemos entón (dentro do directorio **/etc/apache2/sites-available**):

```
cp 000-default.conf server.conf
```

E editamos o arquivo para adaptalo as nosas necesidades:

```
GNU nano 7.2 server.conf
<VirtualHost *:80>
    ServerName server1.rlinares.dedyn.io
    ServerAlias server1
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1
</VirtualHost>
```

Se comprobamos incluíndo o nome en lugar da IP aparentemente funciona



Mais o que está facendo é redirixirnos ao sitio virtual por defecto, pois o noso arquivo de configuración non está habilitado aínda. Isto sabémolo porque no directorio que serve arquivos ao noso VirtualHost non temos ningún arquivo `index.html` con ese contido (pódelo comprobar facendo un `ls` do directorio `/var/www/html/web/server1`).

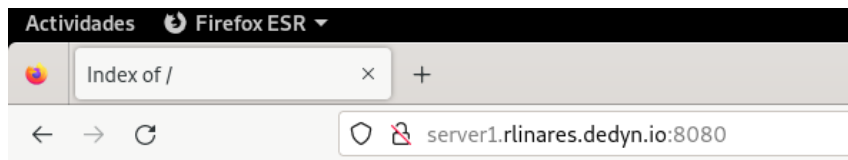
Deste xeito activamos o VirtualHost mediante o comando

a2ensite server.conf








Despois pídennos reiniciar apache2: **service apache2 reload**

```
root@85bd1bffa5008:/etc/apache2/sites-available# a2ensite server.conf
Enabling site server.
To activate the new configuration, you need to run:
    service apache2 reload
root@85bd1bffa5008:/etc/apache2/sites-available# service apache2 reload
Reloading Apache httpd web server: apache2.
root@85bd1bffa5008:/etc/apache2/sites-available#
```

Agora se recargamos a páxina vemos que efectivamente



Index of /

Name	Last modified	Size	Description
 a.html	2023-11-16 22:28	2	
 c.html	2023-11-16 22:28	2	
 dous.html	2023-11-17 14:32	5	
 fotos/	2023-11-16 23:30	-	
 privado/	2023-11-17 00:02	-	
 prueba/	2023-11-16 22:47	-	
 un.html	2023-11-17 14:32	3	

Apache/2.4.57 (Debian) Server at server1.rlinares.dedyn.io Port 8080

se corresponde co contido do directorio establecido como DocumentRoot

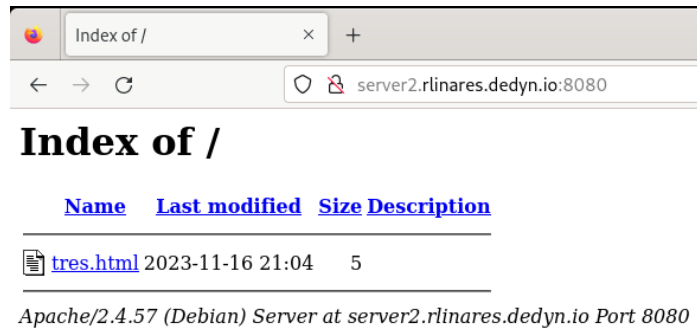
```
root@85bd1bff5008:/var/www/html/web/server1# ls
a.html c.html dous.html fotos privado prueba un.html
```

Temos que incluir agora outro que faga referencia a server2. Aproveitamos para isto o mesmo arquivo.

```
Ficheiro Editar Ver Buscar Terminal Axuda
GNU nano 7.2 server.conf *
<VirtualHost *:80>
    ServerName server1.rlinares.dedyn.io
    ServerAlias server1
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1
</VirtualHost>

<VirtualHost *:80>
    ServerName server2.rlinares.dedyn.io
    ServerAlias server2
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server2
</VirtualHost>
```

Lembrar sempre que fagamos cambios facer un **service apache2 reload**



Que como podemos comprobar se corresponde co contido no directorio establecido como DocumentRoot

```
root@85bd1bfff5008:/var/www/html/web/server2# ls
tres.html
```

Xa temos creados os nosos sitios virtuais cos seus correspondentes DocumentRoot.

Como podemos ver temos acceso a todos os arquivos e directorios, e podemos ver un índice ao acceder a un directorio con contido. Isto débese a que no arquivo de configuración global **apache2.conf** (**/etc/apache2/apache2.conf**) xa temos incluídas estas opcións para o directorio **/var/html** e todos os que están por debaixo del.

```
GNU nano 7.2 apache2.conf
<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

Options Indexes inclúe o listado do contido no directorio e **Require all granted** da permisos para o acceso.

No seguinte punto da tarefa pídesenos: “**Tomando un dos dous servidores virtuais, varía o DirectoryIndex (a nivel do directorio do DocumentRoot), para que teña valores, como a.html, b.html e c.html. Crea varios cartafoles dentro dese, e varía o nome dos ficheiros que alí existen.**”

Vamos a escoller o servidor virtual server1, en cuxo directorio seleccionado no DocumentRoot temos os arquivos a.html e c.html.

Deste modo dentro do VirtualHost correspondente imos a incluír o seguinte, co fin de estableces certas condicións a nivel do directorio.

```

GNU nano 7.2 server.conf *
<VirtualHost *:80>

    ServerName server1.rlinares.dedyn.io
    ServerAlias server1
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1

    <Directory /var/www/html/web/server1>
        DirectoryIndex a.html b.html c.html
    </Directory>

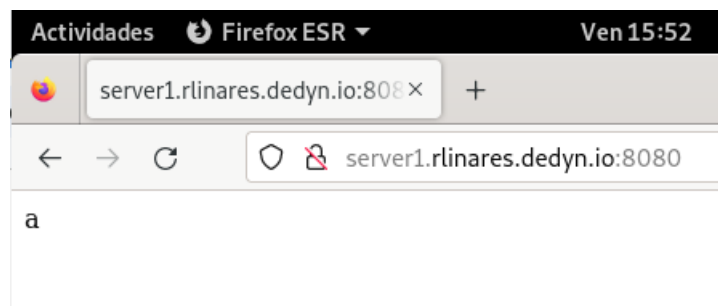
</VirtualHost>

```

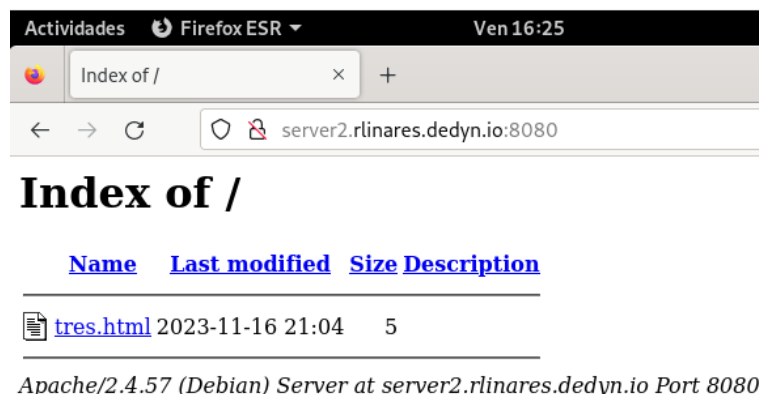
O que fai o **DirectoryIndex** é establecer a lista de recursos a atopar cando o cliente solicita un índice do directorio especificando unha / ao final do nome do directorio.

Pódense poñer varias, pero devolverase a primeira que se atope. Se ningunha existe, e está activada a directiva **Indexes** o servidor xerará un listado do contido do directorio.

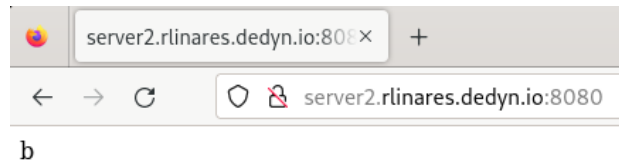
No noso caso:



Se incluísemos isto no VirtualHost do server2 (esto non se pide na tarefa, pero para ver o que ocorre):



Vemos que ao non ter ningún arquivo que se corresponda con incluídos, devolve o índice. Se incluísemos no seu directorio algún arquivo **a.html**, **b.html** ou **c.html**



b

No punto 3 da tarefa: “**Monta outro directorio con volumes en /srv, e fai que esté accesible mediante unha ruta dentro do espazo web.**”

O que se pretende neste apartado é que ao facer unha consulta **http://server1.rlinares.dedyn.io:8080/proba** se chegue a /srv

```
GNU nano 7.2 server.conf *
<VirtualHost *:80>
    ServerName server1.rlinares.dedyn.io
    ServerAlias server1
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory /var/www/html/web/server1>
        DirectoryIndex a.html b.html c.html
    </Directory>

    <Directory /srv>
        Options Indexes
        Require all granted
    </Directory>
    Alias "/proba" "/srv"

</VirtualHost>
```



Se quixéramos que se fixera unha redirección de xeito que cando se faga unha petición a **http://ip/portal**, se redirixa á páxina **http://edu.xunta.gal** (esto non se pide no exercicio) temos que incluír

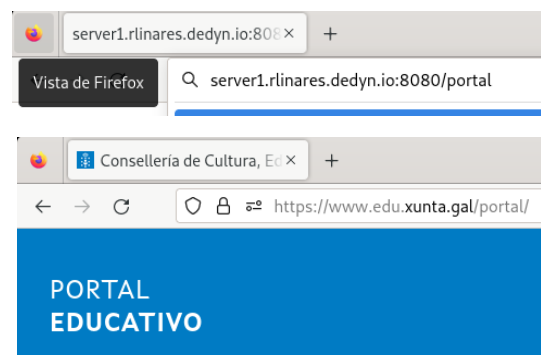
```
GNU nano 7.2 server.conf *
<VirtualHost *:80>
    ServerName server1.rlinares.dedyn.io
    ServerAlias server1
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory /var/www/html/web/server1>
        DirectoryIndex a.html b.html c.html
    </Directory>

    <Directory /srv>
        Options Indexes
        Require all granted
    </Directory>
    Alias "/proba" "/srv"
    Redirect "/portal" "http://edu.xunta.gal"

</VirtualHost>
```



Isto estámolo a facer todo a nivel VirtualHost, de xeito que de facer **/portal** noutro que non sexa server1 non funcionaría. Para iso teríamos que incluílo no arquivo de configuración global e nese caso o redireccionamento funcionaría independentemente de poñer **http://192.168.56.80:8080/portal** , **http://server2.rlinares.dedyn.io:8080/portal** ou **http://server2.rlinares.dedyn.io:8080/portal**

No apartado 4 da tarefa: **“No directorio fotos do DocumentRoot, so queremos que se poidan e visualizar fotos (escolle 6 tipos de arquivo). O resto de extensións denegaranse.”**

No directorio **/var/www/html/web/server1** hai un directorio de fotos que inclúe 5 tipos de arquivos diferentes. Deste modo aínda que no enunciado se nos pide escoller 6 farémolo con 3.

No directorio fotos temos os seguintes arquivos.

```
root@56f252b3193d:/var/www/html/web/server1/fotos# ls
foto1.jpg foto2.jpeg logo.png plano.jpg spock-vulcan.gif star-trek.webp
```

Vamos a permitir que se poidan visualizar únicamente arquivos jpg, gif e png, e denegar todos os demais.

```
<Directory /var/www/html/web/server1/fotos>
    <FilesMatch ".*">
        Order Deny,Allow
        Deny from all
    </FilesMatch>
    <FilesMatch "\.(jpg|gif|png)$">
        Order Allow,Deny
        Allow from all
    </FilesMatch>
</Directory>
</VirtualHost>
```

Order Deny,Allow: establece a orde na que se avalían as regras. Neste caso, primeiro aplícanse as regras de denegar e despois as de permitir. É dicir, por defecto, denégase o acceso a menos que se especifique explicitamente que está permitido.

Order Allow,Deny: tratase do caso contrario ao anterior.

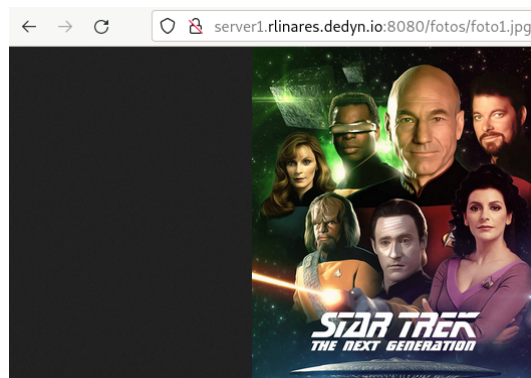
Como neste caso tan só se denega ou permite exclusivamente en cada caso, non é preciso incluír o **Order** e bastaría incluír:

```

<Directory /var/www/html/web/server1/fotos>
    <FilesMatch ".*">
        Deny from all
    </FilesMatch>
    <FilesMatch "\.(jpg|gif|png)$">
        Allow from all
    </FilesMatch>
</Directory>

```

Deste xeito, se intentamos acceder a un arquivo dos permitidos



Mentres que accedendo a un dos non permitidos:



Se o que quixeramos fose denegar algúns e permitir todos os demais (non se nos pide na tarefa)

```

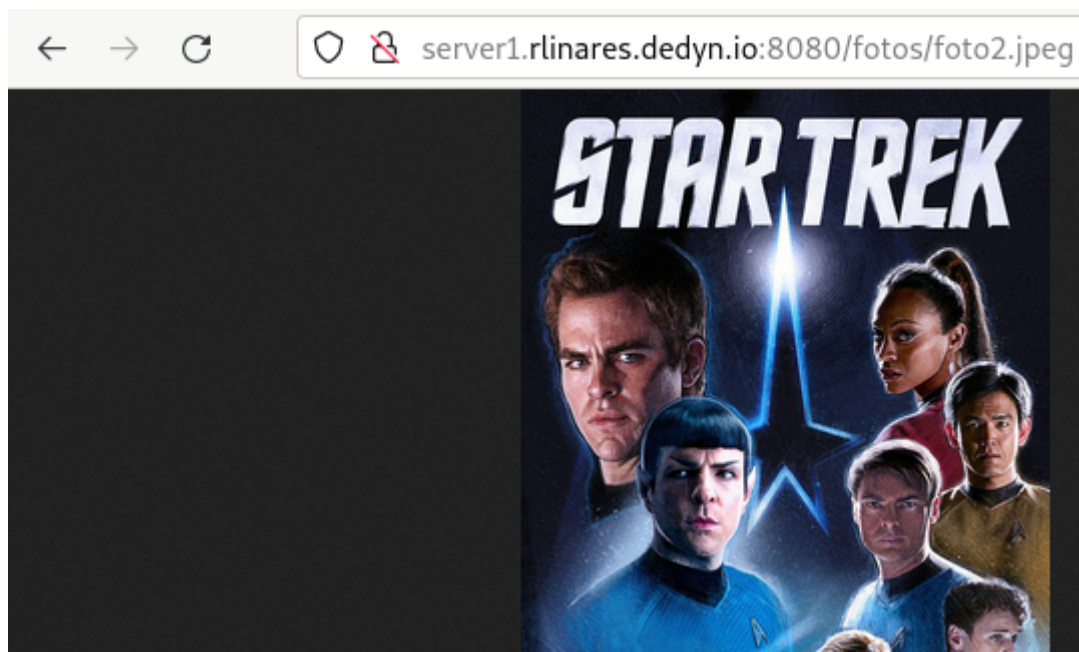
GNU nano 7.2 server.conf

<Directory /var/www/html/web/server1/privado>
    IndexIgnore *.php
</Directory>

<Directory /var/www/html/web/server1/fotos>
    <FilesMatch "\.(jpg|gif|png)$">
        Require all denied
    </FilesMatch>
</Directory>
</VirtualHost>

```

Entón non nos permitirían visualizar a imaxe foto1.jpg, pero si a foto2.jpeg.



No seguinte apartado: **“No directorio privado, amosarás un listado dos ficheiros que ali existan. Os ficheiros .php, non deberán aparecer no listado”**

```
root@85bd1b5f5008:/var/www/html/web/server1/privado# ls
l.html  o.php  r.php  t.txt
```

Estos son os ficheiros que se atopan no directorio

Se queremos que non aparezan no listado debemos incluír:

```

GNU nano 7.2 server.conf *
<VirtualHost *:80>
    ServerName server1.rlinares.dedyn.io
    ServerAlias server1
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory /var/www/html/web/server1>
        DirectoryIndex a.html b.html c.html
    </Directory>

    <Directory /srv>
        Options Indexes
        Require all granted
    </Directory>
    Alias "/proba" "/srv"
    Redirect "/portal" "http://edu.xunta.gal"

    <Directory /var/www/html/web/server1/privado>
        IndexIgnore *.php
    </Directory>

</VirtualHost>

```

Entón os arquivos **.php** non aparecen no listado, pero si se pode acceder aos arquivos

Name	Last modified	Size	Description
Parent Directory	-	-	-
l.html	2023-11-17 00:02	2	
t.txt	2023-11-17 00:02	2	

Apache/2.4.57 (Debian) Server at server1.rlinares.dedyn.io Port 8080

Name	Last modified	Size	Description
Parent Directory	-	-	-
l.html	2023-11-17 00:02	2	
t.txt	2023-11-17 00:02	2	

Apache/2.4.57 (Debian) Server at server1.rlinares.dedyn.io Port 8080

No apartado 6 da tarefa dise: **“Fai o mesmo que todo o anteriores, no server2.lan pero empregando ficheiros .htaccess”**

Os ficheiros **.htaccess** permiten incluír modificacións a nivel directorio sen necesidade de modificar o arquivo de configuración.

Para iso téñense que permitir as actualizacións, incluíndo no directorio **AllowOverride All** como se indica a continuación (poderíanse permitir só certas actualizacións mediante estes arquivos se en lugar de poñer **All** se especificase unha concreta).

```

GNU nano 7.2 server.conf
</Directory>

</VirtualHost>

<VirtualHost *:80>
    ServerName server2.rlinares.dedyn.io
    ServerAlias server2
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server2

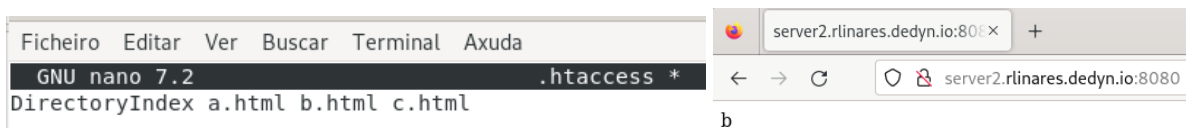
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory /var/www/html/web/server2>
        AllowOverride All
    </Directory>

</VirtualHost>

```

Se queremos facer o solicitado no punto 2 creamos un arquivo **.htaccess** no directorio en cuestión co contido correspondente e ao facer reload



(Incluín un arquivo b.html)

Se queremos facer o correspondente ao apartado 3 temos que ter en conta que estabamos a empregar a directiva **Alias**. Non todas as directivas se poden empregar en todos os contextos (server config, virtual host, directory, .htaccess) polo que é convinte consultar na documentación oficial:

Alias Directive	
Description:	Maps URLs to filesystem locations
Syntax:	Alias [URL-path] file-path directory-path
Context:	server config, virtual host, directory
Status:	Base
Module:	mod_alias

De modo que se pode ver que a directiva **Alias** non se pode empregar nos arquivos **.htaccess**.

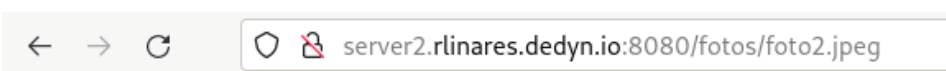
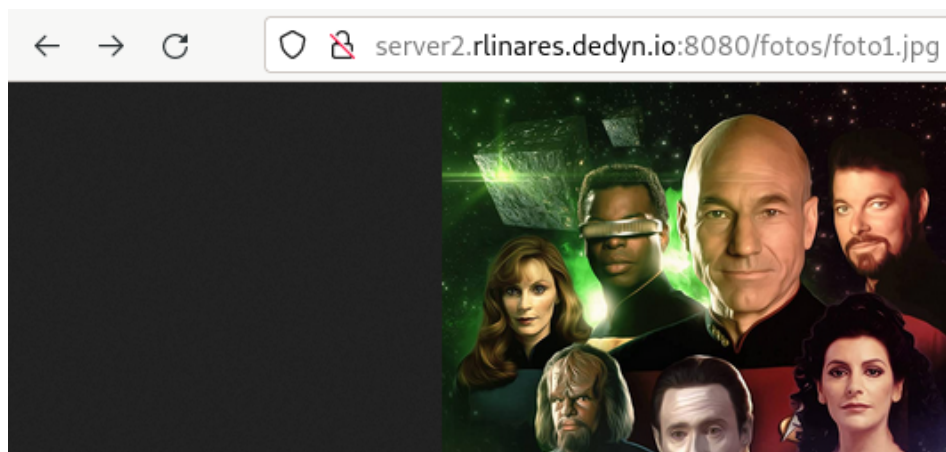
Se queremos facer o solicitado no punto 4 creamos un arquivo **.htaccess** no directorio fotos (**/var/www/html/web/server2/fotos**) co contido correspondente e ao facer reload


```

GNU nano 7.2 .htaccess *
<FilesMatch ".*">
    Order Deny,Allow
    Deny from all
</FilesMatch>

<FilesMatch "\.(jpg|gif|png)$">
    Order Allow,Deny
    Allow from all
</FilesMatch>

```



Forbidden

You don't have permission to access this resource.

Apache/2.4.57 (Debian) Server at server2.rlinares.dedyn.io Port 8080

Para o apartado 5 creamos un arquivo **.htaccess** no directorio privado (/var/www/html/web/server2/privado) co contido correspondente

```

Ficheiro  Editar  Ver  Buscar  Terminal  Axuda
GNU nano 7.2 .htaccess *
IndexIgnore *.php

```

Facer **service apache2 reload**

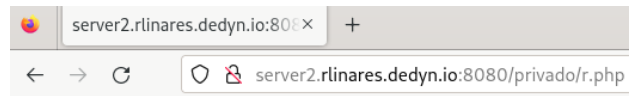
Así, aínda que no directorio hai arquivos **.php**

```

root@85bd1bffa5008:/var/www/html/web/server2/privado# ls
r.php  t.txt

```

Estes non se amosan no índice, pero si se pode acceder a eles



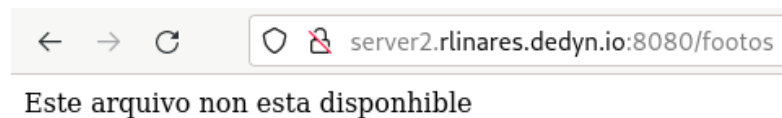
No seguinte apartado, o apartado 7: **“Configura as páxinas de erros personalizados”**

No arquivo de configuración xeral:

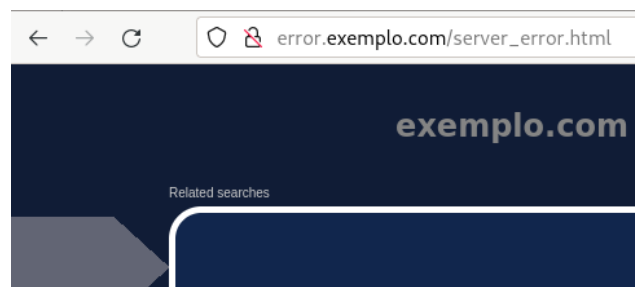
```
GNU nano 7.2                                apache2.conf *
```

```
ErrorDocument 404 "Este arquivo non esta dispoñible"
ErrorDocument 403 http://error.exemplo.com/server_error.html
```

Para o erro 404



Para o erro 403



No referente ao apartado 8: **“Permite que a ruta /interno so se lle permita acceder a un determinado equipo.”**

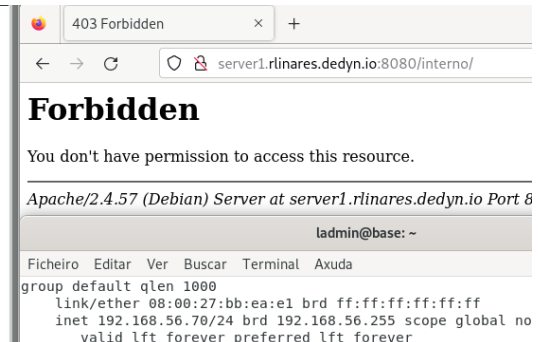
Temos que controlar entón o acceso a unha determinada ruta en función da IP do equipo. Para iso pódese empregar a directiva:

Require ip ip.address

Así, se para o directorio /interno dentro do DocumentRoot do server1 incluímos a directiva anterior indicando unha ip que non corresponde á do cliente en cuestión obtemos o seguinte erro. O cliente debe ser unha máquina distinta á do servidor.

```
<VirtualHost *:80>
    ServerName server1.rlinares.dedyn.io
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1

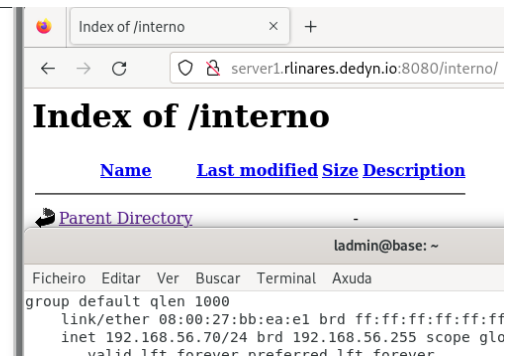
    <Directory /var/www/html/web/server1/interno>
        Require ip 192.168.56.40
    </Directory>
</VirtualHost>
```



Mentres que se permitimos que este poida acceder:

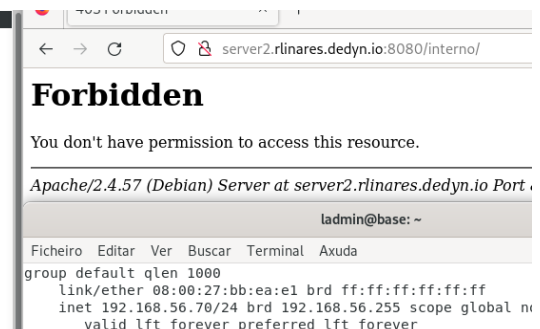
```
<VirtualHost *:80>
    ServerName server1.rlinares.dedyn.io
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/server1

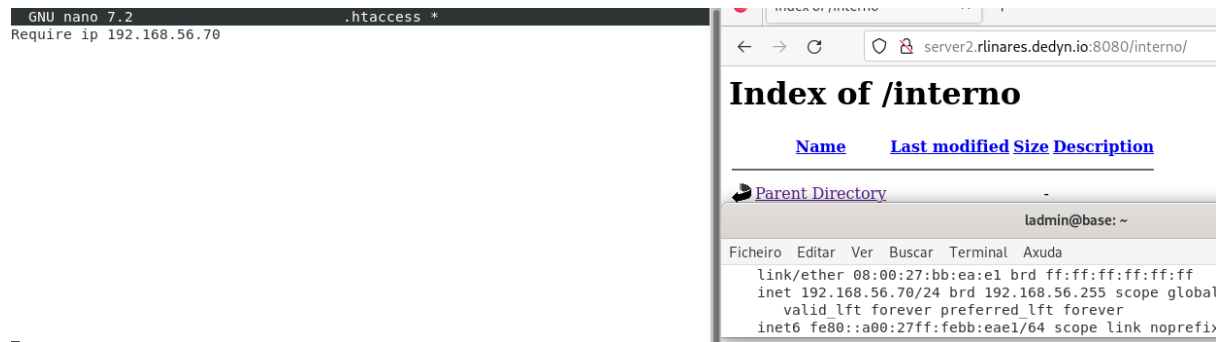
    <Directory /var/www/html/web/server1/interno>
        Require ip 192.168.56.70
    </Directory>
</VirtualHost>
```



Isto mesmo se podería facer empregando arquivos **.htaccess** (compróbase no server2 porque xa estaban permitíramos os arquivos **.htaccess**). Vemos que o obtido correspóndese co que se pide.

```
GNU nano 7.2 .htaccess
Require ip 192.168.56.40
```





Se o que quixesemos fose permitir o acceso a todos mesmos a un equipo concreto teriamos que incluír o seguinte:

```
<RequireAll>
```

```
    Require all granted
```

```
    Require not ip ip.address
```

```
</RequireAll>
```

Deste xeito permítese o acceso dende calquera IP excepto a indicada.

No apartado 9 da tarefa: **“Configura un sitio virtual como calquera dos anteriores, pero empregando HTTPS”**.

No arquivo de configuración creamos un sitio virtual que atenda no porto 443 (deberíamos ter asignado no docker-compose.yml un porto do anfitrión ao contedor)

Asignamoslle un novo nome (asociado a mesma IP) e incluímos chaves (certificado dixital e chave privada). Podemos pensar en host virtuais que soporten conexións con protocolos http e https, pero non é aconsellable. É mellor crear un host virtual novo que soporte SSL. Podemos empregar unha parella chave privada e certificado dixital autoasinado xa presentes no noso equipo (como é ssl-cert-snakeoil). Se facemos iso, temos que ter en conta que calquera navegador pode non recoñecer o certificado e será necesario unha excepción. Poderíamos obter un certificado válido para empregar con Apache2 doutro xeito, pero de momento, para comprobar que funciona e ver o proceso, quedámonos con este.

```
<VirtualHost *:443>
    ServerName sitio1.rlinares.dedyn.io
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/web/sitio1

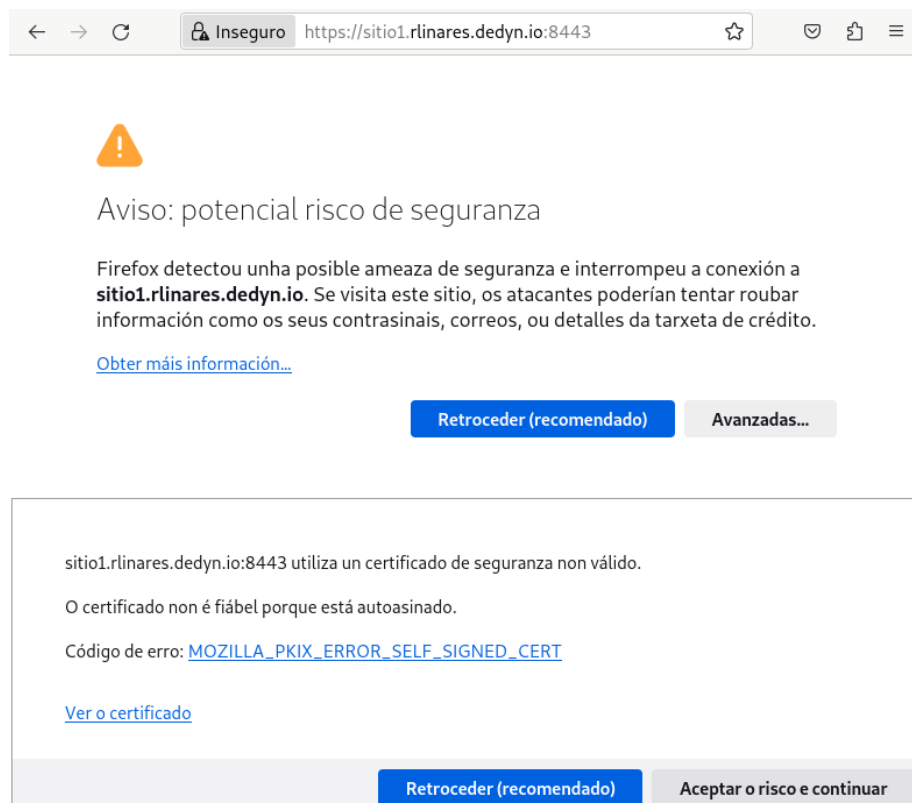
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    SSLEngine on
    SSLCertificateFile "/etc/ssl/certs/ssl-cert-snakeoil.pem
    SSLCertificateKeyFile "/etc/ssl/private/ssl-cert-snakeoil.key

</VirtualHost>
```

O soporte de SSL ven de serie co paquete de Apache2 de Ubuntu/Debian. So é necesario habilitar o módulo **ssl** para empregar todas as vantaxes que prové SSL no noso sistema, empregando para iso **a2enmod ssl**

Como comentamos antes, ao non recoñecer o certificado, apareceranos unha advertencia, e debemos facer unha excepción (**Aceptar o risco e continuar**).




Puidendo empregar agora https



https://sitio1.rlinares.dedyn.io:8443

Index of /

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
-------------	----------------------	-------------	--------------------

 sitio.html	2023-11-20 19:21	7	
--	------------------	---	--

Apache/2.4.57 (Debian) Server at sitio1.rlinares.dedyn.io Port 8443